Some properties of variable length packet shapers

Jean-Yves Le Boudec
<jean-yves.leboudec@epfl.ch>
EPFL-DSC
CH-1015 Lausanne, Switzerland

EPFL-DSC Research Report DSC/2000/037

20 Decembre 2000

Abstract

The min-plus theory of greedy shapers has been developed after Cruz's results on the calculus of network delays. An example of greedy shaper is the buffered leaky bucket controller. The theory of greedy shapers establishes a number of properties; for example, re-shaping keeps original arrival constraints. The existing theory applies in all rigor either to fluid systems, or to packets of constant size such as ATM. For variable length packets, the distortion introduced by packetization affects the theory, which is no longer valid. Chang has introduced the concept of packetizer, which models the effect of variable length packets, and has also developed a max-plus theory of shapers. In this paper, we start with the min-plus theory, and obtain fundamental results on greedy shapers for variable length packets which are not readily explained with the max-plus theory of Chang. We show a fundamental result, namely, the min-plus representation of a packetized greedy shaper. This allows us to prove that, under some assumptions, re-shaping a flow of variable length packets does keep original arrival constraints. However, we show on some examples that if the assumptions are not satisfied, then the property may not hold any more. We also demonstrate the equivalence of implementing a buffered leaky bucket controller based on either virtual finish times or on bucket replenishment.

Keywords: shapers; variable length packets; network calculus; leaky bucket

1 Introduction

We consider some of the problems caused by traffic regulation for flows of variable length packets. While the original work by Cruz in [1] defines a leaky bucket regulator as a system handling variable length packets, the theory of regulators (which we now call " greedy shapers") that was later developed [2, 3, 4, 5] either focused explicitly on flows of constant size packets (namely ATM), or applies only to fluid systems. The theory of greedy shapers is extremely powerful (in its context); it allows to establish a number of invariance or optimality properties; for example, re-shaping keeps original arrival constraints. These properties have been used for example for designing schedulers [6].

For variable length packets, the distortion introduced by packetization affects the theory, which is no longer valid. Chang has introduced in [7] the concept of packetizer, which models the effect of variable length packets. In many cases, packetizers weaken the bounds obtained with a fluid model by one packet size of maximum size, however, as recalled in Section 2.3, in some cases this distortion may be accumulated at every node. Shapers are important in the context of integrated as well as differentiated services; they are known under terms such

as token bucket, or leaky bucket controllers. This motivates us to understand the effect of variable packet sizes on the properties of shapers. We take as starting point the packetizer and study its relationship with optimal shaping.

In this paper, we take a fundamental look at the issue, and obtain both fundamental and practical results. Firstly, we show an input-output representation of greedy shapers for flows of variable size packets (Theorem 4.2). We find conditions under which this representation can be considerably simplified (Theorem 4.1). Secondly (Section 5), we prove that, under some assumptions, re-shaping a flow of variable length packets does keep original arrival constraints. However, we show on some examples that if the assumptions are not satisfied, then the property may not hold any more. Thirdly, we consider two commonly used alternative implementations of leaky bucket controllers, which we call the "virtual finish time" and the "bucket replenishment" implementations. We show that both implementations produce the same packet output (Corollary 4.1).

The paper is organized as follows. Section 2 reviews the state of the art on greedy shapers, recalls the concept of packetizers, as introduced in [7], and gives our notation. Section 3 gives a first theoretical results, which is the basis for practical results derived in the rest of the paper. It shows that in some good cases, the concatenation of a bit-by-bit greedy shaper and a packetizer keeps arrival constraints. Section 4 is the main theory; we introduce the concept of packetized greedy shaper and obtain its input-output characterization. Section 5 examines whether a packetized greedy shaper keeps arrival constraints. The proofs of all theorems are put in appendix.

2 State of the art

2.1 Regulator, Buffered Leaky Bucket Controller and Greedy Shaper

We use the standard terminology and say that a flow is σ-smooth, or has σ has arrival curve, for some function $\sigma(t)$, if the number of bits observed on the flow during a time interval of duration t is $\leq \sigma(t)$. The IETF uses a generic family of arrival curves of the form $\sigma(t) = \min(M + pt, b + rt)$ where M is interpreted as a maximum packet size, p a peak rate, b a burst tolerance and r a sustainable rate [8]. Similarly, a "buffered leaky bucket controller" is a system which forces a flow to be σ-smooth, with $\sigma(t) = \min_{m=1,\ldots,M}(r_m t + b_m)$, while delaying the packets as little as possible. This is traditionally interpreted by saying that the controller observes a set of M fluid buckets, where the mth bucket is of size b_m and leaks at a constant rate r_m. Every bucket receives l_i units of fluid when packet i is released (l_i is the size of packet i). A packet is released as soon as the level of fluid in bucket m allows it, namely, has gone down below $b_m - l_i$, for all m. We will use a variant of this definition later in this paper; for clarity, we say that we have defined now a buffered leaky bucket controller based on "bucket replenishment".

A variant of the buffered leaky bucket controller was studied in a continuous time setting by Cruz in [1] under the name of "(b, r) regulator". The regulator is associated with a bucket of fluid which leaks at a constant rate r; a packet is released as soon as the level of fluid in the bucket does not exceed b. The output of the regulator has a constant rate C, which corresponds to a physical line rate. The output of such a regulator is σ-smooth, with

$$\sigma(t) = rt + b + \frac{1-r}{C} l_{\max} \tag{1}$$

where $l_{\max}$ is the maximum packet size. This variant differs from the standard definition by the term $\frac{1-r}{C} l_{\max}$ and by the fact that it explicitly accounts for a line rate C at the output. As we will see in detail in Section 2.4, the set of outputs of such regulators is not identical with the set of packet flows that are σ-smooth, with σ given by Equation (1). Thus this form of regulator is not exactly a buffered leaky bucket controller.

In [2], Cruz refined the concept in discrete time with constant size packets. It was found in particular that

regulators can be combined to synthesize systems that force a flow to be σ-smooth where σ is any concave and piecewise linear wide-sense increasing function. The optimality of regulators was established, namely, a regulator in discrete time forces its output to be σ-smooth, and does it as early as possible. This was used to show that regulators enjoy some remarkable properties, for example, if a regulated flow is passed through a second regulator, then the final output keeps the arrival curve constraint imposed by the first regulator. In the discrete time setting, the regulator is exactly a buffered leaky bucket controller.

The concept was further generalized independently in [3, 4, 5] under the name of greedy shaper. Given some wide-sense increasing function σ, a greedy shaper takes some flow as input and forces its output to be σ-smooth; it delays the input data in a buffer, whenever sending data would violate the constraint σ, but outputs them as soon as possible. This generalizes the concept of buffered leaky bucket controller, this latter system corresponding to a function σ which is piecewise linear and concave. The theory shows that, without loss of generality, we can assume that $\sigma(0) = 0$ and that σ is sub-additive, namely, $\sigma(s+t) \leq \sigma(s) + \sigma(t)$. Concave, increasing functions are sub-additive, but there are useful sub-additive functions that are not concave (Section 2.2). The cornerstone result is the input/output characterization of greedy shapers. Call $R(t)$ the cumulative input function (namely, the number of bits observed in time interval $[0, t]$) and $R^*(t)$ the output of the greedy shaper, we have

$$R^*(t) = \inf_{0 \leq s \leq t} (\sigma(s) + R(t-s)) = (\sigma \otimes R)(t) \tag{2}$$

The right-handside in the equation is called the min-plus convolution of σ and R, and is usually noted $(\sigma \otimes R)(t)$. The simple, intuitive properties of min-plus convolution are then used in [3, 4, 5]. In this paper, we focus on the property, already mentioned, that greedy shapers keep arrival constraints.

The theory in [2, 3] is for discrete time systems with constant packet sizes, and thus applies without restriction to ATM systems. In contrast, and unlike the original results in [1, 9], the theory of greedy shapers in [4, 5] applies to continuous time and flows with variable packet size, but does not account for packetization effects. In this context, the greedy shaper characterized by Equation (2) outputs a continuous stream of bits, not entire packets; see for example Figure 1. We call it a "bit-by-bit greedy shaper". In some cases, it is a good model: for example, a constant bit rate trunk with rate C is modeled with a bit-by-bit greedy shaper with $\sigma(t) = Ct$ (for $t \geq 0$) (but see also Figure 1). In general, though, regulators used in various packet scheduling methods output entire packets and cannot strictly be modeled with a bit-by-bit greedy shaper, but can be captured by the concept of "packetizer", recalled in Section 2.3.

2.2 Non concave arrival curves

Much attention has been given to concave arrival curves, because they naturally appear with leaky buckets. However, non concave arrival curves are sometimes used in the context of switch dimensioning. As an example, we will use in this paper the "stair function" v_T, defined by

$$v_T(t) = \begin{cases} \lceil \frac{t}{T} \rceil \text{ if } t > 0 \\ 0 \text{ if } t \leq 0 \end{cases}$$

Saying that a flow is v_T-smooth is equivalent to saying that during any window of time T, the flow has at most 1 unit of data. For packets of fixed size, this is equivalent to a spacing condition [10]. Consider for example an ATM switch receiving n connections, each of them perfectly shaped with peak rate T.[1] The aggregate flow is nv_T-smooth; characterizing this aggregate flow with a concave arrival curve would lead to looser dimensioning bounds, see [11] for an example where this is used. In this paper we will briefly mention the impact of non-concave arrival curves, mainly because they help understand the fundamentals.

[1] A more realistic example would use $v_{T,\tau}(t) = \lceil \frac{t+\tau}{T} \rceil$ for $t > 0$ in order to account for cell delay variation tolerance

2.3 The packetizer

The packetizer was introduced by Chang in [7] as a means to model packetization effect. We say that a sequence $L = (L(0) = 0, L(1), L(2), ...)$ is a "sequence of cumulative packet lengths" if it is wide sense increasing and

$$l_{\min} := \inf_n \{L(n+1) - L(n)\}$$

is positive. We interpret $L(n) - L(n-1)$ as the length of the nth packet.

Now for any sequence of cumulative packet lengths L we define function P^L by

$$P^L(x) = \sup_{n \in \mathbb{N}} \{L(n) 1_{\{L(n) \leq x\}}\} \tag{3}$$

Intuitively, $P^L(x)$ is the largest cumulative packet length which is entirely contained in x. Function P^L is right-continuous; if R is right-continuous, then so is $P^L(R(t))$. For example, if all packets have unit length, then $L(n) = n$ and for $x > 0$: $P^L(x) = \lfloor x \rfloor$. An equivalent characterization of P^L is

$$P^L(x) = L(n) \Longleftrightarrow L(n) \leq x < L(n+1) \tag{4}$$

Note also that, with our definition:

$$x - l_{\max} < P^L(x) \leq x \tag{5}$$

Then Chang [7] defines an "L-packetizer" as the system which transforms an input $R(t)$ into $P^L(R(t))$. Figure 1 illustrates how a constant bit rate trunk is modeled by a greedy shaper followed by a packetizer.

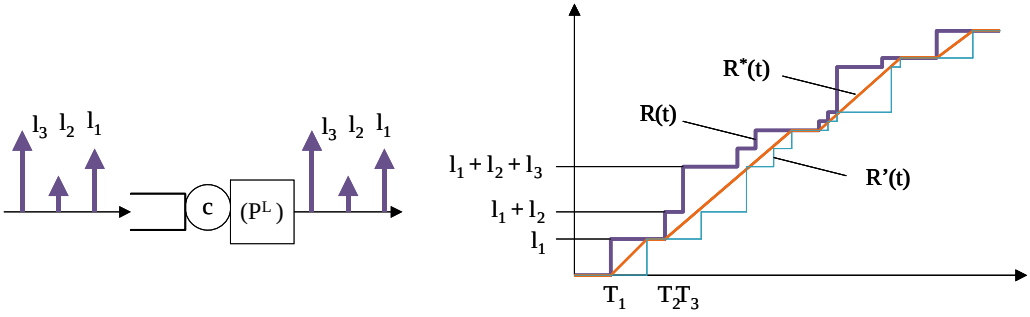

Figure 1: A trunk of constant bit rate C, viewed as the concatenation of a greedy shaper and a packetizer. The input is $R(t)$, the output of the greedy shaper is $R^*(t) = R \otimes \lambda_C(t)$, with $\lambda_C(t) = Ct1_{\{t \geq 0\}}$. The final output is $R'(t) = P^L(R^*(t))$.

We also say that a flow $R(t)$ is L-packetized if $P^L(R(t)) = R(t)$ for all t.

A packetizer introduces some distortion to the results in [3, 4, 5]. Consider Figure 1; we know that a greedy shaper keeps arrival constraints, thus if R is σ-smooth for some σ, then so is R^*. However, this is not true for R'. Consider the following example, described in [12]. Assume that $\sigma(t) = l_{\max} + rt$ with $r < C$. Assume that the input flow $R(t)$ sends a first packet of size $l_1 = l_{\max}$ at time $T_1 = 0$, and a second packet of size l_2 at time $T_2 = \frac{l_2}{r}$. Thus the flow R is indeed σ-smooth. The departure time for the first packet is $T_1' = \frac{l_{\max}}{C}$. Assume that the second packet l_2 is small, namely $l_2 < \frac{r}{C} l_{\max}$; then the two packets are sent back-to-back and thus the departure time for the second packet is $T_2' = T_1' + \frac{l_2}{C}$. Now the spacing $T_2' - T_1'$ is less than $\frac{l_2}{r}$, thus the second packet is not conformant, in other words, R' is not σ-smooth. Note that this example is not possible

if all packets have the same size. This is a first example of the relationship between packetizer and shaper. We will find general results in Section 3.

Bounds based on the maximum packet size can easily be derived. The following items come mostly from [7]. Consider a system (*bit-by-bit system*) with L-packetized input R and bit-by-bit output R^*, which is then L-packetized to produce a final packetized output R'. We call *combined system* the system which maps R into R'. Assume both systems are first-in-first-out and lossless.

1. The *per-packet delay* for the combined system is $\sup_i (T'_i - T_i)$, where T_i, T'_i are the arrival and departure time for the ith packet. It is equal to the maximum virtual delay for the bit-by-bit system.

2. Call B^* the maximum backlog for the bit-by-bit system and B' the maximum backlog for the combined system. We have $B^* \leq B' \leq B^* + l_{\max}$.

3. Assume that the bit-by-bit system offers to the flow a a maximum service curve γ and a minimum service curve β. The combined system offers to the flow a maximum service curve γ and a minimum service curve β' given by $\beta'(t) = [\beta(t) - l_{\max}]^+$. Thus packetizing weakens the service curve guarantee by one maximum packet length. For example, if a system offers a rate-latency service curve with rate R, then appending a packetizer to the system has the effect of increasing the latency by $\frac{l_{\max}}{R}$. The rate-latency service curve with rate R and latency T is defined by $S(t) = R(t-T)^+$. It is commonly used to model a generic scheduler.

4. If some flow $S(t)$ has $\sigma(t)$ as arrival curve, then $P^L(S(t))$ has $\sigma(t) + l_{\max} 1_{\{t>0\}}$ as arrival curve. For Figure 1 this tells us that the final output R' has $\sigma'(t) = \sigma(t) + l_{\max} 1_{t>0}$ as arrival curve, which is consistent with our observation that R' is not σ-smooth, even though R^* is. We will see in Section 4 that there is a stronger result, in relation with the concept of "packetized greedy shaper".

Item 1 says that appending a packetizer to a node does not increase the packet delay at this node. However, packetization does increase the end-to-end delay. Consider the concatenation of the theoretical GPS node, with guaranteed rate R [13] and an L-packetizer. Assume this system receives a flow of variable length packets. This models a theoretical node which would work as a GPS node but is constrained to deliver entire packets. This is not very realistic, but is sufficient to explain one important effect of packetizers.

In the fluid model, the GPS node offers a rate latency service curve with some rate R and 0 latency (in the simplest case in [13]). By application of item 3, we find that the combined node offers a rate-latency service curve with rate R and latency $T = \frac{l_{\max}}{R}$. Now concatenate m such identical nodes, as illustrated on Figure 2.

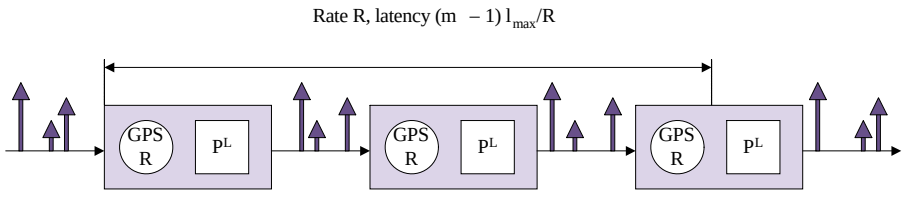

Figure 2: The concatenation of several GPS fluid nodes with packetized outputs

By application of standard results, the end-to-end service curve is the rate latency-function with rate R and latency $T = m\frac{l_{\max}}{R}$. However, for the computation of the end-to-end delay bound, we need to take into account item 1, which tells us that we can forget the last packetizer. Thus, a bound on end-to-end delay is obtained by considering that the end-to-end path offers a service curve equal to the latency-function with rate R and latency

$T_0 = (m-1)\frac{l_{\max}}{R}$. For example, if the original input flow is constrained by one leaky bucket of rate r and bucket pool of size b, then an end-to-end delay bound is

$$\frac{b + (m-1)l_{\max}}{R} \tag{6}$$

A generalization of the reasoning above to the family of "guaranteed rate" schedulers (which contains PGPS) is straightforward [14]: (6) is replaced by $\frac{b+(m-1)l_{\max}}{R} + mv$ where v characterizes the scheduler lateness, compared to GPS (for example, for PGPS, $v = \frac{M}{C}$ where M is the maximum packet size across all flows and C is the total server rate). In [7] there is a slightly weaker formula, with $ml_{\max}$ instead of $(m-1)l_{\max}$.

We see on this example that the additional latency introduced by one packetizer is indeed of the order of one packet length; however, this effect is multiplied by the number of hops, roughly speaking. This illustrates that the distortion introduced when taking into account packetization effects can be more than one packet length, which motivates us to do the theory in the rest of this paper.

2.4 The max-plus theory of shapers

A dual approach to account for variable length packets is introduced in [7]. It consists in replacing the definition of σ-smoothness, mentioned above, by the concept of g-regularity. Consider a flow of variable length packets, with cumulative packet length L and call T_i the arrival epoch for the ith packet. The flow is said to be g-regular if $T(j) - T(i) \geq g(L(j) - L(i))$ for all packet numbers $i \leq j$. A theory is then developed with concepts similar to the greedy shaper. The theory uses max-plus convolution instead of min-plus convolution. The (b, r) regulator of Cruz is a shaper in this theory, whose output is g-regular, with $g(x) = \frac{(x-b)^+}{r}$.

Note that this theory does not exactly correspond to the usual concept of leaky bucket controllers. More specifically, there is not an exact correspondence between the set of flows that are g-regular on one hand, and that are σ-smooth on the other. We explain why on an example. Consider the set of flows that are g-regular, with $g(x) = \frac{x}{r}$. The minimum arrival curve we can put on this set of flows is $\sigma(t) = rt + l_{\max}$ [7]. But conversely, if a flow is σ-smooth, we cannot guarantee that it is g-regular. Indeed, the following sequence of packets is a flow which is σ-smooth but not g-regular: the flow has a short packet (length $l_1 < l_{\max}$) at time $T_1 = 0$, followed by a packet of maximum size $l_{\max}$ at time $T_2 = \frac{l_1}{r}$. In fact, if a flow is σ-smooth, then it is g'-regular, with $g'(x) = \frac{(x-l_{\max})^+}{r}$. Nonetheless, this theory is a very elegant and powerful complement to the min-plus theory in [3, 4, 5].

In this paper we focus on the properties of regulators, and on the min-plus theory rather than the max-plus. As a consequence, our results apply to the usual definition of leaky bucket controllers and arrival constraints as given in Section 2.1.

3 A relation between greedy shaper and packetizer

We have seen that appending a packetizer to a greedy shaper weakens the arrival curve property of the output. There is however a case where this is not true. This case is important for the results in Section 4, but also has practical applications of its own.

Theorem 3.1. *Consider a sequence L of cumulative packet lengths and call (P^L) the L-packetizer. Consider a sub-additive function σ with $\sigma(0) = 0$, and assume that*

$$\begin{cases} \textit{There exists a sub-additive function } \sigma_0 \textit{ and a number } l \geq l_{\max} \textit{ such that} \\ \sigma(t) = \sigma_0(t) + l1_{t>0} \end{cases} \tag{7}$$

and call (σ) the greedy shaper with shaping curve σ. For any input, the output of the concatenation[2] *$(P^L)(\sigma)(P^L)$ is σ-smooth.*

In practical terms, the theorem is used as follows. Consider an L-packetized flow, pass it through a greedy shaper with shaping curve σ; and packetize the output; then the result is σ-smooth (assuming that σ satisfies condition in Equation (7)). Figure 3 illustrates the theorem.

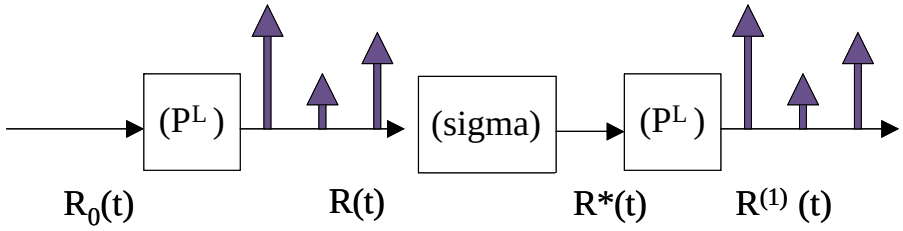

Figure 3: Theorem 3.1 says that $R^{(1)}$ is σ-smooth.

Note that in general the output of $(P^L)(\sigma)$ is *not* L-packetized, even if σ satisfies the condition in the theorem (finding a counter-example is simple and is left to the reader's enjoyment). Similarly, if the input to $(\sigma)(P^L)$ is not L-packetized, then the output is not σ-smooth, in general.

The theorem could also be rephrased by saying that, under condition in Equation (7)

$$(P^L)(\sigma)(P^L) = (P^L)(\sigma)(P^L)(\sigma)$$

since the two above operators always produce the same output.

Discussion of Condition in Equation (7) Condition Equation (7) is satisfied in practice if σ is concave and $\sigma(0^+) \geq l_{\max}$, where $\sigma(0^+) = \inf_{t>0} \sigma(t)$ is the limit to the right of σ at 0. This occurs for example if the shaping curve is defined by the conjunction of leaky buckets, all with bucket size at least as large as the maximum packet size.

This also sheds some light on example in Figure 1: the problem occurs because the shaping curve λ_C does not satisfy the condition.

The alert reader will ask herself whether a sufficient condition for Equation (7) to hold is that σ is sub-additive and $\sigma(0^+) \geq l_{\max}$. Unfortunately, the answer is no. Consider for example the stair function $\sigma = l_{\max} v_T$. We have $\sigma(0^+) = l_{\max}$ but if we try to decompose σ into $\sigma(t) = \sigma_0(t) + l 1_{t>0}$ we must have $l = l\max$ and $\sigma_0(t) = 0$ for $t \in (0, T]$; if we impose that σ_0 is sub-additive, the latter implies $\sigma_0 = 0$ which is not compatible with Equation (7).

Buffered Leaky Bucket Controller based on Virtual Finish Times Theorem 3.1 gives us an alternative implementation of the buffered leaky bucket controller. We build it as the concatenation of a buffered leaky bucket controller operating bit-by-bit and a packetizer. We compute the output time for the last bit of a packet (= finish time) under the bit-by-bit leaky bucket controller, and release the entire packet instantly at this finish time. If each bucket pool is at least as large as the maximum packet size then Theorem 3.1 tells us that the final output satisfies the leaky bucket constraints. Note that this implementation differs from the buffered leaky bucket controller based on bucket replenishment introduced in Section 2.1. In the former, during a period

[2] We use the notation $(\sigma)(P^L)$ to denote the consecutive operation of the two operators, with (σ) applied first.

where, say, bucket m only is full, fragments of a packet are virtually released at rate r_m, bucket m remains full, and the (virtual) fragments are then re-assembled in the packetizer; in the latter, if a bucket becomes full, the controller waits until it empties by at least the size of the current packet. Thus we expect that the level of fluid in both systems is not the same, the former being an upper bound. We will see however in Section 4 that both implementations are equivalent.

Counter-example If we consider non-concave arrival curves, then we can find an arrival curve σ which does satisfy $\sigma(t) \geq l_{\max}$ for $t > 0$ but which does not satisfy Equation (7). In such a case, the conclusion of Theorem 3.1 may not hold in general. Figure 4 shows an example where the output $R^{(1)}$ is not σ-smooth, when σ is a stair function.

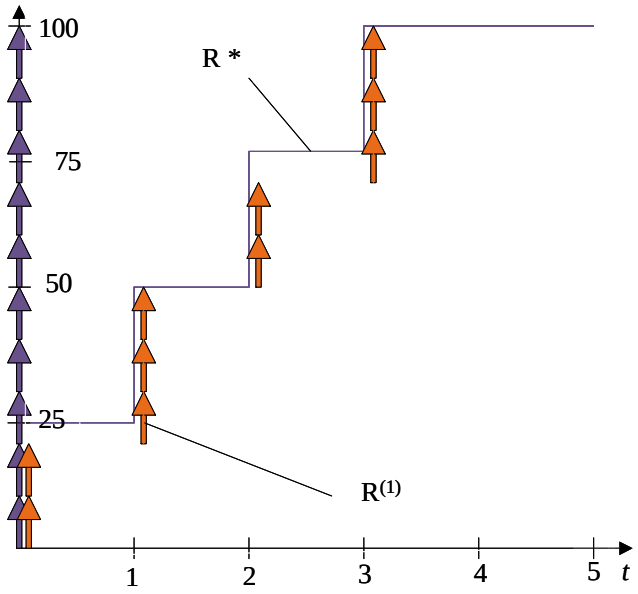

Figure 4: A counter example for Theorem 3.1. A burst of 10 packets of size equal to 10 data units arrive at time $t = 0$, and $\sigma = 25v_1$. The greedy shaper emits 25 data units at times 0 and 1, which forces the packetizer to create a burst of 3 packets at time 1, and thus $R^{(1)}$ is not σ-smooth.

4 Packetized Greedy Shaper

We now give a more fundamental look at the issue of packetized shaping. We introduce the following definition as a natural abstraction of the buffered leaky bucket controller.

Definition 4.1. *[Packetized Greedy Shaper] Consider an input sequence of packets. Call L the cumulative packet lengths. We call* packetized shaper*, with shaping curve σ, a system which forces its output to have σ as arrival curve and be L-packetized. We call* packetized greedy shaper *a packetized shaper which delays the input packets in a buffer, whenever sending a packet would violate the constraint σ, but outputs them as soon as possible.*

The buffered leaky bucket controller defined in Section 2.1 is clearly a packetized greedy shaper with $\sigma(t) = \min_{m=1,\ldots,M}(r_m t + b_m)$.

If some bucket size b_m is less than the maximum packet size, then it is never possible to output a packet: all packets remain stuck in the packet buffer, and the output is 0. In general, if $\sigma(0^+) < l_{\max}$ then the the packetized greedy shaper blocks all packets for ever. Thus, for practical cases, we have to assume that the arrival curve σ has a discontinuity at the origin at least as large as one maximum packet size.

If Equation (7) is satisfied, then the realization of a packetized shaper is simple:

Theorem 4.1 (Realization of packetized Greedy Shaper). *Consider a sequence L of cumulative packet lengths and a sub-additive function σ with $\sigma(0) = 0$. Assume that σ satisfies the condition in Equation (7). Consider only inputs that are L packetized. Then the packetized greedy shaper for σ and L can be realized as the concatenation of the bit-by-bit greedy shaper with shaping curve σ and the L-packetizer.*

We have seen that the condition in the theorem is satisfied in particular if σ is concave and $\sigma(0^+) \geq l_{\max}$, for example if the shaping curve is defined by the conjunction of leaky buckets, all with bucket size at least as large as the maximum packet size. This shows the following.

Corollary 4.1. *For L-packetized inputs, the implementations of buffered leaky bucket controllers based on bucket replenishment and virtual finish times are equivalent.*

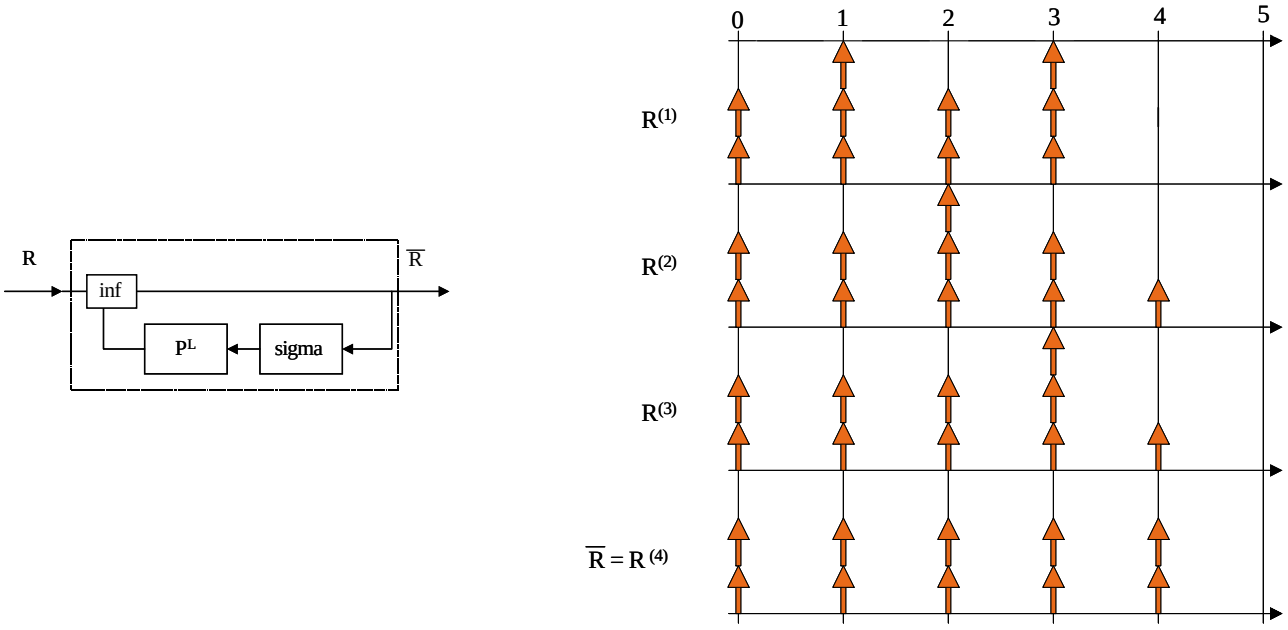

Figure 5: Representation of the output of the packetized greedy shaper (left) and example of output (right). The data are the same as with Figure 4.

If we relax Equation (7) then the construction of the packetized greedy shaper is more complex:

Theorem 4.2 (I/O characterisation of packetized greedy shapers). *Consider a packetized greedy shaper with shaping curve σ and cumulative packet length L. Assume that σ is sub-additive and $\sigma(0) = 0$. The output $\overline{R}(t)$ of the packetized greedy shaper is given by*

$$\overline{R} = \inf\left\{R^{(1)}, R^{(2)}, R^{(3)}, ...\right\} \tag{8}$$

with $R^{(1)}(t) = P^L((\sigma \otimes R)(t))$ and $R^{(i)}(t) = P^L((\sigma \otimes R^{(i-1)})(t))$ for $i \geq 2$.

Figure 5 illustrates the theorem, and shows the iterative construction of the output on one example. Note that this example is for a shaping function that does not satisfy Equation (7). Indeed, otherwise, we know from Theorem 4.1 that the iteration stops at the first step, namely, $\overline{R} = R^{(1)}$ in that case. We can also check for example that if $\sigma(t) = Ct$ for $t \geq 0$ (thus the condition $\sigma(0^+) < l_{\max}$ is satisfied) then the result of Equation (8) is 0.

5 Does a packetized greedy shaper keep arrival constraints ?

Figure 6 shows a counter-example, namely a variable length packet flow which has lost its initial arrival curve constraint after traversing a packetized greedy shaper.

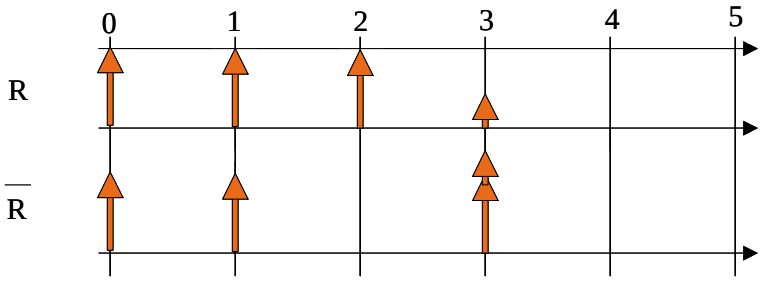

Figure 6: The input flow is shown above; it consists of 3 packets of size 10 data units and one of size 5 data units, spaced by one time unit. It is σ-smooth with $\sigma = 10u_{1,0}$. The bottom flow is the output of the packetized greedy shaper with $\sigma = 25u_{3,0}$. The output has a burst of 15 data units packets at time 3. It is σ-smooth but *not* α-smooth.

However, if arrival curves are defined by leaky buckets, we have a positive result.

Theorem 5.1 (Conservation of concave arrival constraints). *Assume an L-packetized flow with arrival curve α is input to a packetized greedy shaper with cumulative packet length L and shaping curve σ. Assume that α and σ are concave with $\alpha(0^+) \geq l_{\max}$ and $\sigma(0^+) \geq l_{\max}$. Then the output flow is still constrained by the original arrival curve α.*

Thus, we have proven that the conservation property found in [9] for (b, r)-regulators holds for the more usually accepted definition of buffered leaky bucket controller. However, we have also found that, unlike the results for constant size packets in [3, 4, 5], we cannot, in general, extend this property to any arbitrary arrival curve.

6 Conclusion

We have extended the min-plus theory of variable length packet shaping and shown some fundamental results which account for the distortion introduced by packetization affects. Our main theoretical result is the min-plus representation of a packetized greedy shaper. This allows us to prove that, under some assumptions, re-shaping a flow of variable length packets does keep original arrival constraints. However, we show on some examples that if the assumptions are not satisfied, then the property may not hold any more. We also demonstrate the equivalence of implementing a buffered leaky bucket controller based on either virtual finish times or on bucket replenishment. It remains to be seen how other structural properties of greedy shapers, which are valid for constant size packets, are affected by variability in packet size. If some deviations are found, it

will be important to know which of them remain contained to one maximum packet size, and,in contrast, which of them accumulate over network paths.

References

[1] R.L. Cruz, "A calculus for network delay, part i: Network elements in isolation," *IEEE Trans. Inform. Theory, vol 37-1*, pp. 114–131, January 1991.

[2] R.L. Cruz, "Quality of service guarantees in virtual circuit switched networks," *IEEE JSAC*, pp. 1048–1056, August 1995.

[3] C.S. Chang, "On deterministic traffic regulation and service guarantee: A systematic approach by filtering," *IEEE Transactions on Information Theory*, vol. 44, pp. 1096–1107, August 1998.

[4] J.-Y. Le Boudec, "Application of network calculus to guaranteed service networks," *IEEE Transactions on Information Theory*, vol. 44, pp. 1087–1096, May 1998.

[5] R. Agrawal, R. L. Cruz, C. Okino, and R. Rajan, "Performance bounds for flow control protocols," *IEEE/ACM Transactions on Networking (7) 3*, pp. 310–323, June 1999.

[6] R. L. Cruz, "Sced+ : Efficient management of quality of service guarantees," in *IEEE Infocom'98, San Francisco*, March 1998.

[7] C.S. Chang, *A filtering theory for Communication Networks*, Springer Verlag, 1999.

[8] R. Guérin S. Shenker, C. Partridge, "Specification of guaranteed quality of service," Septembre 1997, RFC 2702, IETF.

[9] R.L. Cruz, "A calculus for network delay, part ii: Network analysis," *IEEE Trans. Inform. Theory, vol 37-1*, pp. 132–141, January 1991.

[10] I. Chlamtac, A. Faragó, H. Zhang, and A. Fumagalli, "A deterministic approach to the end-to-end analysis of packet flows in connection oriented networks," *IEEE/ACM transactions on networking*, vol. (6)4, pp. 422–431, 08 1998.

[11] J.-Y. Le Boudec and G. Hebuterne, "Comment on a deterministic approach to the end-to-end analysis of packet flows in connection oriented network," *IEEE/ACM Transactions on Networking*, February 2000.

[12] R. Guérin and V. Pla, "Aggregation and conformance in differentiated service networks – a case study," Tech. Rep. Research Report, U Penn, http://www.seas.upenn.edu:8080/ guerin/publications/aggreg.pdf, August 2000.

[13] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single node case," *IEEE/ACM Trans. Networking, vol 1-3*, pp. 344–357, June 1993.

[14] P. Goyal, S. S. Lam, and H. Vin, "Determining end-to-end delay bounds in heterogeneous networks," in *5th Int Workshop on Network and Op. Sys support for Digital Audio and Video, Durham NH*, April 1995.

[15] J.-Y. Le Boudec and P. Thiran, *High Performance Networks for Multimedias Applications*, chapter 9, Network Calculus using Min/Plus System Theory,, `http://ica1www.epfl.ch/PS_files/chap9Klu.ps`, pp. 136–159, Kluwer, 1999.

7 Appendix: Proof of Theorems

7.1 Proofs of Theorem 3.1

We use the notation in Figure 3. We want to show that $R^{(1)}$ is σ-smooth. We have $R^* = R \otimes \sigma$. Consider now some arbitrary s and t with $s < t$. From the definition of min-plus convolution, for all $\epsilon > 0$, there exists some $u \leq s$ such that

$$(R \otimes \sigma)(s) \geq R(u) + \sigma(s-u) - \epsilon \tag{9}$$

Now consider the set E of $\epsilon > 0$ such that we can find one $u < s$ satisfying the above equation. Two cases are possible: either 0 is an accumulation point for E (case 1) , or not (case 2).

Consider case 1; there exists a sequence (ϵ_n, s_n), with $\lim_{n\to+\infty} \epsilon_n = 0$, $s_n < s$ and

$$(R \otimes \sigma)(s) \geq R(s_n) + \sigma(s - s_n) - \epsilon_n$$

Now since $s_n \leq t$:

$$(R \otimes \sigma)(t) \leq R(s_n) + \sigma(t - s_n)$$

Combining the two:

$$(R \otimes \sigma)(t) - (R \otimes \sigma)(s) \leq \sigma(t - s_n) - \sigma(s - s_n) + \epsilon_n$$

Now $t - s_n > 0$ and $s - s_n > 0$ thus

$$\sigma(t - s_n) - \sigma(s - s_n) = \sigma_0(t - s_n) - \sigma_0(s - s_n)$$

We have assumed that σ_0 is sub-additive. Now $t \geq s$ thus

$$\sigma_0(t - s_n) - \sigma_0(s - s_n) \leq \sigma_0(t - s)$$

we have thus shown that, for all n

$$(R \otimes \sigma)(t) - (R \otimes \sigma)(s) \leq \sigma_0(t - s) + \epsilon_n$$

and thus

$$(R \otimes \sigma)(t) - (R \otimes \sigma)(s) \leq \sigma_0(t - s)$$

Now from Equation (5), it follows that

$$R^{(1)}(t) - R^{(1)}(s) \leq \sigma_0(t - s) + l_{\max} \leq \sigma(t - s)$$

which ends the proof for case 1.

Now consider case 2. There exists some ϵ_0 such that for $0 < \epsilon < \epsilon_0$, we have to take $u = s$ in Equation (9). This implies that

$$(R \otimes \sigma)(s) = R(s)$$

Now R is L-packetized by hypothesis. Thus

$$R^{(1)}(s) = P^L((R \otimes \sigma)(s)) = P^L(R(s)) = R(s) = (R \otimes \sigma)(s)$$

thus

$$R^{(1)}(t) - R^{(1)}(s) = P^L((R \otimes \sigma)(t) - (R \otimes \sigma)(s) \leq (R \otimes \sigma)(t) - (R \otimes \sigma)(s)$$

now the $R \otimes \sigma$ has σ as arrival curve thus finally

$$R^{(1)}(t) - R^{(1)}(s) \leq \sigma(t - s)$$

which ends the proof for case 2. □

7.2 Proof of Theorem 4.1

Call $R(t)$ the packetized input; the output of the bit-by-bit greedy shaper followed by a packetizer is $R^{(1)}(t) = P^L(R \otimes \sigma)(t))$. Call $\overline{R}(t)$ the output of the packetized greedy shaper. We have $\overline{R} \leq R$ thus $\overline{R} \otimes \sigma \leq R \otimes \sigma$ and thus

$$P^L(\overline{R} \otimes \sigma) \leq P^L(R \otimes \sigma)$$

But $\overline{R}$ is σ-smooth, thus $\overline{R} \otimes \sigma = \overline{R}$, and is L-packetized, thus $P^L(\overline{R} \otimes \sigma) = \overline{R}$. Thus the former inequality can be rewritten as $\overline{R} \leq R^{(1)}$. Conversely, from Theorem 3.1, $R^{(1)}$ is also σ-smooth and L-packetized. The definition of the packetized greedy shaper implies that $\overline{R} \geq R^{(1)}$ (for a formal proof, see Lemma 7.1) thus finally $\overline{R} = R^{(1)}$. □

7.3 Proof of Theorem 4.2

The proof a direct application of Lemma 7.1 and Lemma 7.2. It can also be obtained by application of the general method in [15]. □

Lemma 7.1. *Consider a sequence L of cumulative packet lengths and a sub-additive function σ with $\sigma(0) = 0$. Among all flows $x(t)$ such that*

$$\begin{cases} x \leq R \\ x \text{ is } L\text{-packetized} \\ x \text{ has } \sigma \text{ as arrival curve} \end{cases} \tag{10}$$

there is one flow $\overline{R}(t)$ which upper-bounds all. It is given by Equation (8).

Proof: If x is a solution, then it is straightforward to show by induction on i that $x(t) \leq R^{(i)}(t)$ and thus $x \leq \overline{R}$. The difficult part is now to show that $\overline{R}$ is indeed a solution. We need to show that the three conditions in Equation (10) hold. Firstly, $R^{(1)} \leq R(t)$ and by induction on i, $R^{(i)} \leq R$ for all i; thus $\overline{R} \leq R$.

Secondly, consider some fixed t; $R^{(i)}(t)$ is L-packetized for all $i \geq 1$. Let $L(n_0) := R^{(1)}(t)$. Since $R^{(i)}(t) \leq R^{(1)}(t)$, $R^{(i)}(t)$ is in the set $\{L(0), L(1), L(2), ..., L(n_0)\}$. This set is finite, thus, $\overline{R}(t)$, which is the infimum of elements in this set, has to be one of the $L(k)$ for $k \leq n_0$. This shows that $\overline{R}(t)$ is L-packetized, and this is true for any time t.

Thirdly, we have, for all i

$$\overline{R}(t) \leq R^{(i+1)}(t) = P^L((\sigma \otimes R^{(i)})(t)) \leq (\sigma \otimes R^{(i)})(t)$$

thus

$$\overline{R} \leq \inf_i (\sigma \otimes R^{(i)})$$

From Lemma 7.2, $\inf_i(\sigma \otimes R^{(i)}) = \sigma \otimes \overline{R}$ thus

$$\overline{R} \leq \sigma \otimes \overline{R}$$

which shows the third condition. □

Lemma 7.2 (Convolution by a fixed function is upper-semi-continuous). *Consider some function $\sigma(t)$ with $\sigma(0) = 0$. Consider also a sequence of functions $x_n(t)$ such that $x_{n+1}(t) \leq x_n(t)$ for all t and n and call $x = \inf_n x_n$. Then*

$$\inf_n (\sigma \otimes x_n) = \sigma \otimes x$$

Proof: By definition of min-plus convolution, we have, for all $t \geq 0$:

$$(\sigma \otimes x_n)(t) = \inf_{s \in [0,t]} (\sigma(s) + x_n(t-s))$$

thus

$$\begin{aligned} \inf_n (\sigma \otimes x_n)(t) &= \inf_{s \in [0,t], n \in \mathbb{N}} [\sigma(s) + x_n(t-s)] \\ &= \inf_{s \in [0,t]} \{\inf_{n \in \mathbb{N}} [(\sigma(s) + x_n(t-s)]\} \\ &= \inf_{s \in [0,t]} \{\sigma(s) + \inf_{n \in \mathbb{N}} [x_n(t-s)]\} \\ &= \inf_{s \in [0,t]} [\sigma(s) + x(t-s)] \\ &= (\sigma \otimes x)(t) \end{aligned}$$

□

7.4 Proof of Theorem 5.1

We use the notation in Theorem 3.1. Call $R(t)$ the input to the packetized greedy shaper. By application of Theorem 4.1, since α satisfies Equation (7), there exists some R_0 such that

$$R_0 \to (P^L)(\alpha)(P^L) \to R$$

where $(P^L)(\alpha)(P^L)$ means the successive operation of the three operators. By the same token, the output $\overline{R}$ of the packetized greedy shaper satisfies

$$R_0 \to (P^L)(\alpha)(P^L)(P^L)(\sigma)(P^L) \to \overline{R}$$

Now (P^L) is idempotent, thus

$$R_0 \to (P^L)(\alpha)(P^L)(\sigma)(P^L) \to \overline{R}$$

From Theorem 3.1, we have $(P^L)(\alpha)(P^L) = (P^L)(\alpha)(P^L)(\alpha)$, thus

$$R_0 \to (P^L)(\alpha)(P^L)(\alpha)(\sigma)(P^L) \to \overline{R}$$

Note that $(\alpha)(\sigma) = (\alpha \otimes \sigma)$, thus

$$R_0 \to (P^L)(\alpha)(P^L)(\alpha \otimes \sigma)(P^L) \to \overline{R}$$

Now from our hypothesis, $\alpha \otimes \sigma = \alpha \wedge \sigma$ and thus $\alpha \otimes \sigma$ satisfies Equation (7). Thus

$$R_0 \to (P^L)(\alpha)(P^L)(\alpha \otimes \sigma)(P^L)(\alpha \otimes \sigma) \to \overline{R}$$

which shows that $\overline{R}$ is $\alpha \otimes \sigma$-smooth, and thus α-smooth.

□